|  |  |  |
| --- | --- | --- |
| **Politechnika Świętokrzyska w Kielcach**  **Wydział Elektroniki, Automatyki i Informatyki** | | |
| projekt: **Programowanie w języku C2** | | |
| Rok studiów:  **2** | Temat:  **Gra szachy** | Grupa: **2ID12A**  Piotr Michałek  Izabela Bożęcka |

1. **Ogólny opis projektu:**

Mieliśmy za zadanie jak wskazuje nazwa projektu stworzyć grę szachy. Gra bazuje na bibliotece służącej do pisania gier 2D, SFML. Zawiera ona obiektowy interfejs dlatego też zdecydowaliśmy się na użycie właśnie tej biblioteki, ze względu na przyjemne jej użytkowanie. W projekcie został użyty silnik służący do generowania ruchów komputera.

1. **Informacje na temat funkcjonalności projektu:**

Projekt umożliwia grę w szachy. Składa się z menu głównego w którym możemy zadecydować czy chcemy rozpocząć nową grę, zobaczyć instrukcję poruszania się bierek czy też chcemy zakończyć rozgrywkę. Jeżeli w menu głównym wybierzemy nową grę zostaniemy przekierowani na szachownicę w celu rozpoczęcia rozgrywki.

1. **Informacje na temat sposobu uruchomienia oraz obsługi projektu:**

Projekt uruchamiamy za pomocą klawisza F9. Po jego wciśnięciu zostanie uruchomione menu główne po którym możemy się poruszać za pomocą przycisków

góra i dół. Jeżeli zaznaczymy w menu jakąś opcję i wciśniemy klawisz enter, wtedy w

zależności od wybranej opcji, rozpoczniemy nowa grę, zobaczymy instrukcję poruszania się bierek lub zakończymy grę. W przypadku gdy chcielibyśmy zobaczyć instrukcję, bezpośrednio z niej również istnieje możliwość rozpoczęcia nowej gry, wystarczy ze za pomocą klawiszy góra, dół najedziemy na „Instrukcja poruszania się bierek’’ i wciśniemy enter. Sama gra obsługiwana jest za pomocą myszy. Istnieją trzy możliwe wersje rozgrywki, komputer vs komputer, gracz vs gracz lub gracz vs komputer. Jeżeli chcemy zobaczyć pojedynek komputera z komputerem wystarczy, że będziemy naciskać spację, a bierki same będą się poruszać. Jeżeli chcemy sami zagrać z komputerem, wtedy chcąc poruszyć się jakąś bierką musimy najechać na nią myszką wcisnąć lewy przycisk myszy przytrzymać i upuścić na interesujące nas pole (drag and drop). Ruch komputera zostanie wykonany po naszym ruchu jeśli naciśniemy klawisz spacji. Możemy również zagrać z żywą osobą wtedy jedna osoba będzie grać białymi a druga czarnymi bierkami. Poruszanie odbywa się tak samo jak w przypadku gry z komputerem, czyli za pomocą myszki. Mamy również możliwość cofnięcia ruchu, w przypadku pomyłki, wystarczy wcisnąć przycisk Backspace.

1. **Informacje na temat stworzonych klas, metod, funkcji**

Gra bazuje na dwóch klasach, jednej o nazwie Game odpowiedzialnej za grę oraz Menu odpowiedzialnej za menu. W klasie Game znajduje się pięć publicznych metod. Pierwsza z nich to metoda typu string o nazwie toChessNote jako parametr przyjmuje obiekt typu Vector2f. Jest wykorzystywana w funkcji gra() w celu wyliczenia starej i nowej pozycji bierki, którą gracz się poruszył. Kolejną metodą jest toCoord, jako parametry przyjmuje dwie zmienne typu char, jest wykorzystywana w funkcjach move() oraz gra(), wykorzystywana jest do zapamiętania starej i nowej pozycji bierki po wykonaniu ruchu. Metoda move() jako parametr przyjmuje zmienną typu string,

Na początku tworzone są dwa obiekty typu Vector2f w których zapamiętuje starą i nową pozycję bierki. Następnie tworzy pętlę for w której sprawdza warunek, czy aktualna pozycja każdej bierki jest równa nowej pozycji. Jeżeli tak jest to każda bierka jest ustawiona na odpowiedniej pozycji. Analogiczna pętla for oraz warunek jest stworzony dla starej pozycji. Metoda wczytaj\_pozycje() nie przyjmuje parametrów ani niczego nie zwraca. Służy do wczytania pozycji poszczególnych bierek na szachownicy, oraz wewnątrz niej następuję wywołanie funkcji move() w pętli for co umożliwia cofnięcie ruchu. Metoda gra() nie przyjmuje żadnych parametrów ani niczego nie zwraca. W niej toczy się cała akcja gry, znajduje się w niej pętla główna gry wraz ze wszystkimi niezbędnymi wywołaniami wyżej wymienionych funkcji. W klasie Menu stworzony jest konstruktor, destruktor oraz cztery publiczne metody. Konstruktor jako parametry przyjmuje dwie zmienne typu float odpowiadające szerokości i wysokości okna oraz tworzy w tym oknie menu główne ładując czcionkę z pliku ustawiając jej wielkość oraz kolor. Metoda instrukcja jest podobna w swojej budowie do konstruktora. Jako parametry przyjmuje dwie zmienne typu float na szerokość i wysokość okna tworząc w tym oknie instrukcję poruszania się bierek.

Metoda rysuj() jako parametr pobiera obiekt typu RenderWindow, niczego nie zwraca, służy do rysowania menu głównego lub instrukcji w zależności od miejsca w którym została wywołana. Metoda ruch\_gora() nie przyjmuje żadnych parametrów, ani niczego nie zwraca, służy do poruszania się w górę po menu głównym. Jeżeli naciśnięta zostanie strzałka w górę wtedy następuje zmiana koloru opcji powyżej z czarnego na czerwony oraz następuje dekrementacja zmiennej zaznaczenie. Metoda ruch\_dol() nie przyjmuje żadnych parametrów, służy do poruszania się w dół po menu głównym. Jeżeli naciśnięta zostanie strzałka w dół wtedy następuje zmiana koloru opcji poniżej z czarnego na czerwony oraz następuje inkrementacja zmiennej zaznaczenie. Procedura ConnectToEngine() jako parametr pobiera wskaźnik typu char, umożliwia połączenie się z silnikiem odpowiedzialnym za generowanie ruchów komputera podczas rozgrywki. Funkcja typu string o nazwie GetNextMove() jako parametr przyjmuje zmienną typu string oznaczającą pozycje bierek na szachownicy, sprawdza ruchy które może wykonać komputer, zwraca ruch który komputer może zrobić lub error jeżeli nie można zrobić żadnego ruchu. Procedura CloseConnection() nie pobiera żadnych parametrów, służy do zamknięcia połączenia z silnikiem generującym ruchy komputera. W funkcji głównej znajduję się przede wszystkim blok instrukcji switch który umożliwia nam poruszanie się po menu i w zależności od wybranej opcji uruchomienie gry, instrukcji bądź wyjście z gry.

1. **Informacje na temat ilości pracy włożonej przez poszczególnych członków zespołu.**

Ciężko stwierdzić konkretnie, kto jaką wykonał pracę, gdyż generalnie podczas prac nad projektem pracowaliśmy zawsze razem. Można stwierdzić, że nakład pracy włożonej przez członków zespołu jest identyczny ponieważ nad wszystkim co znajduje się w naszym projekcie myśleliśmy wspólnie.